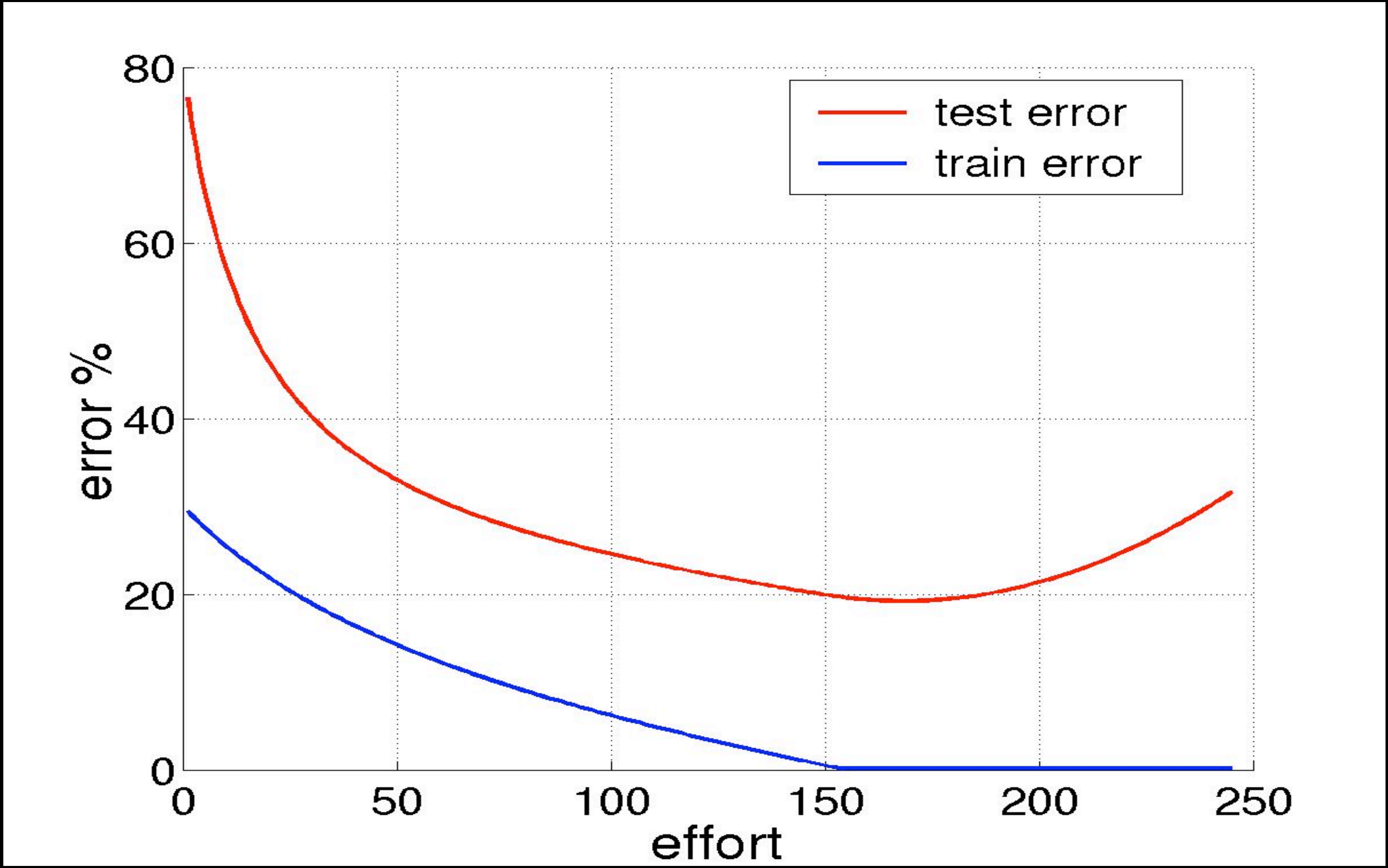
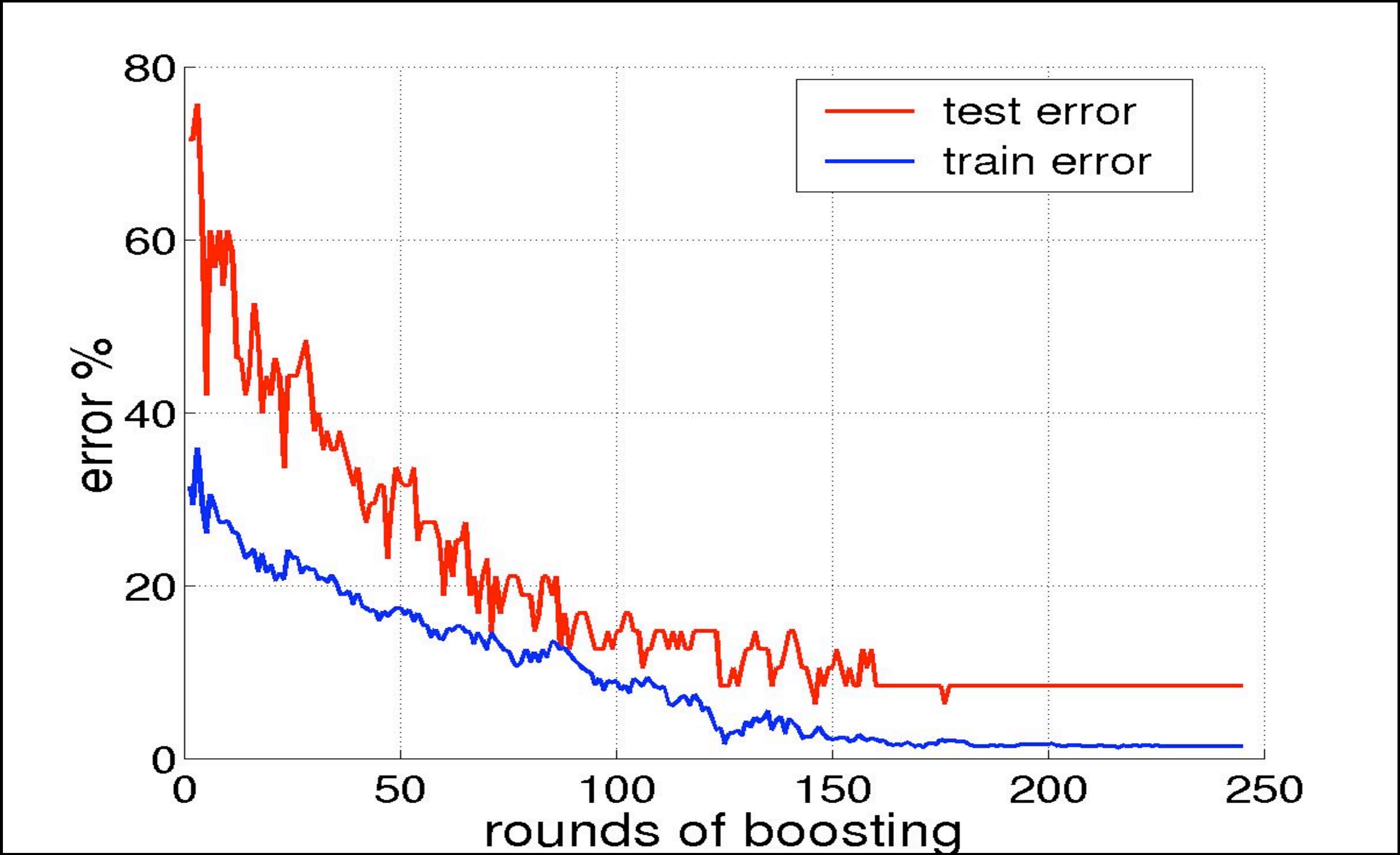
The AdaBoost Algorithm

A typical learning curve



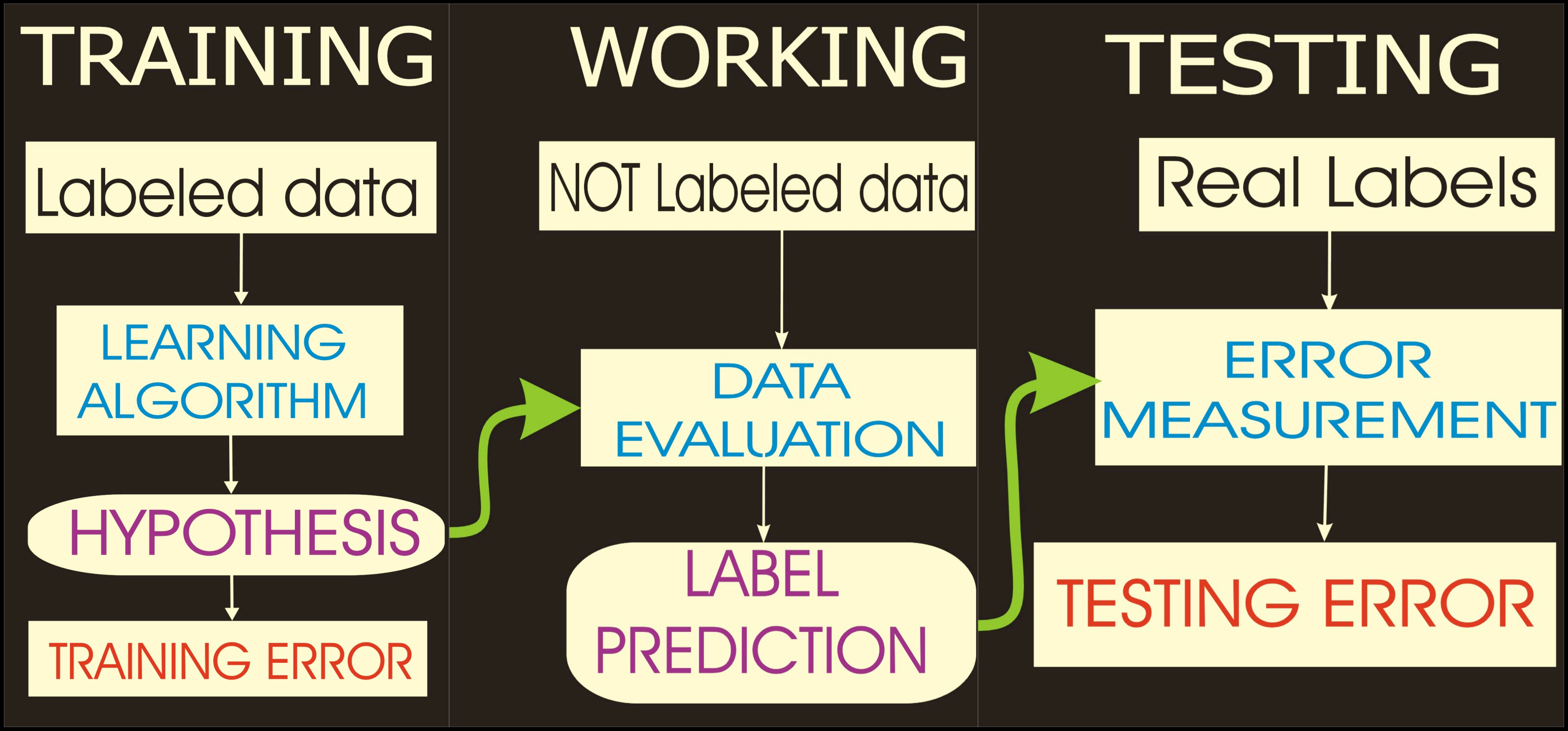
…and a boosting one



this lecture

- boosting and on-line learning
- AdaBoost algorithm
- extensions
- applications

supervised learning



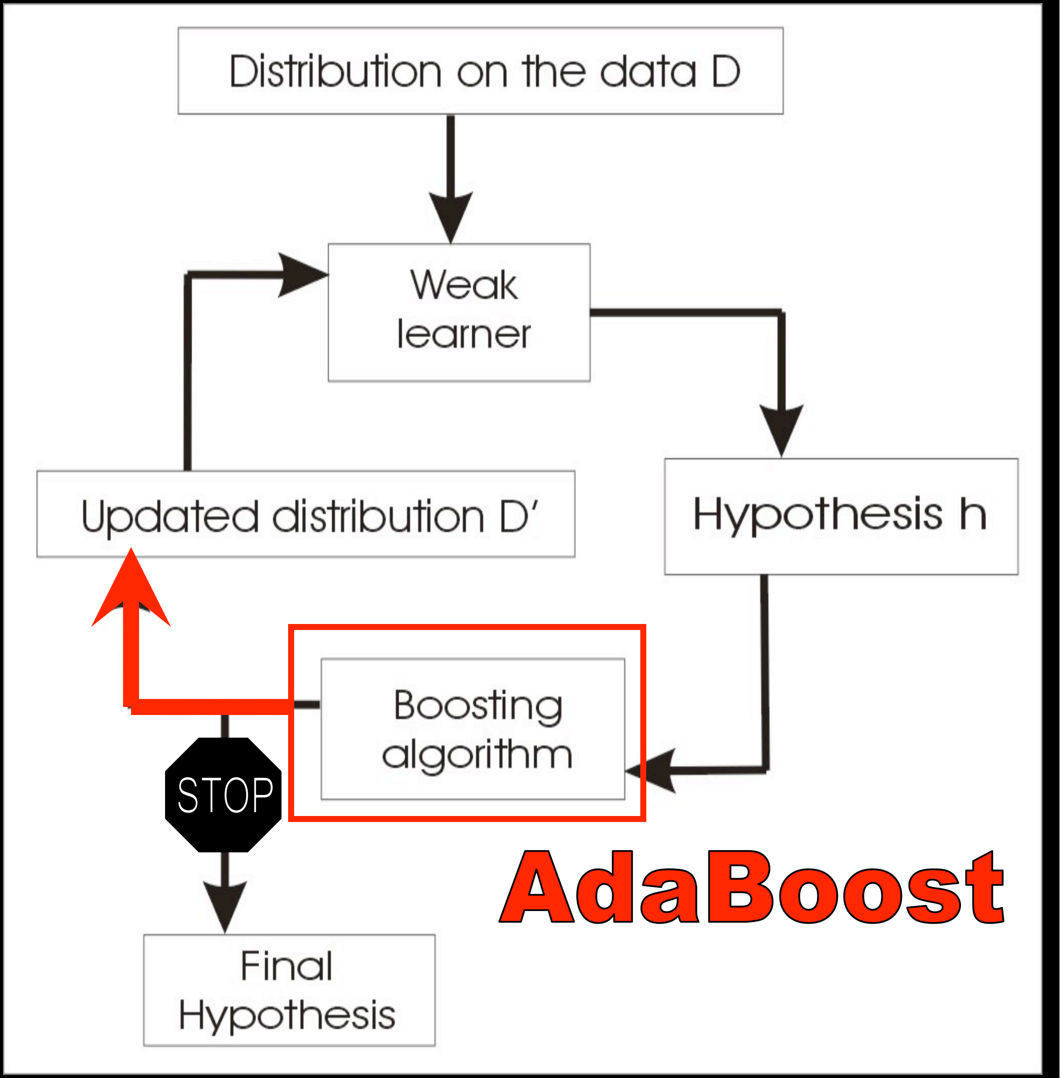
- Data model

$$X = \{(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_m, y_m)\}$$

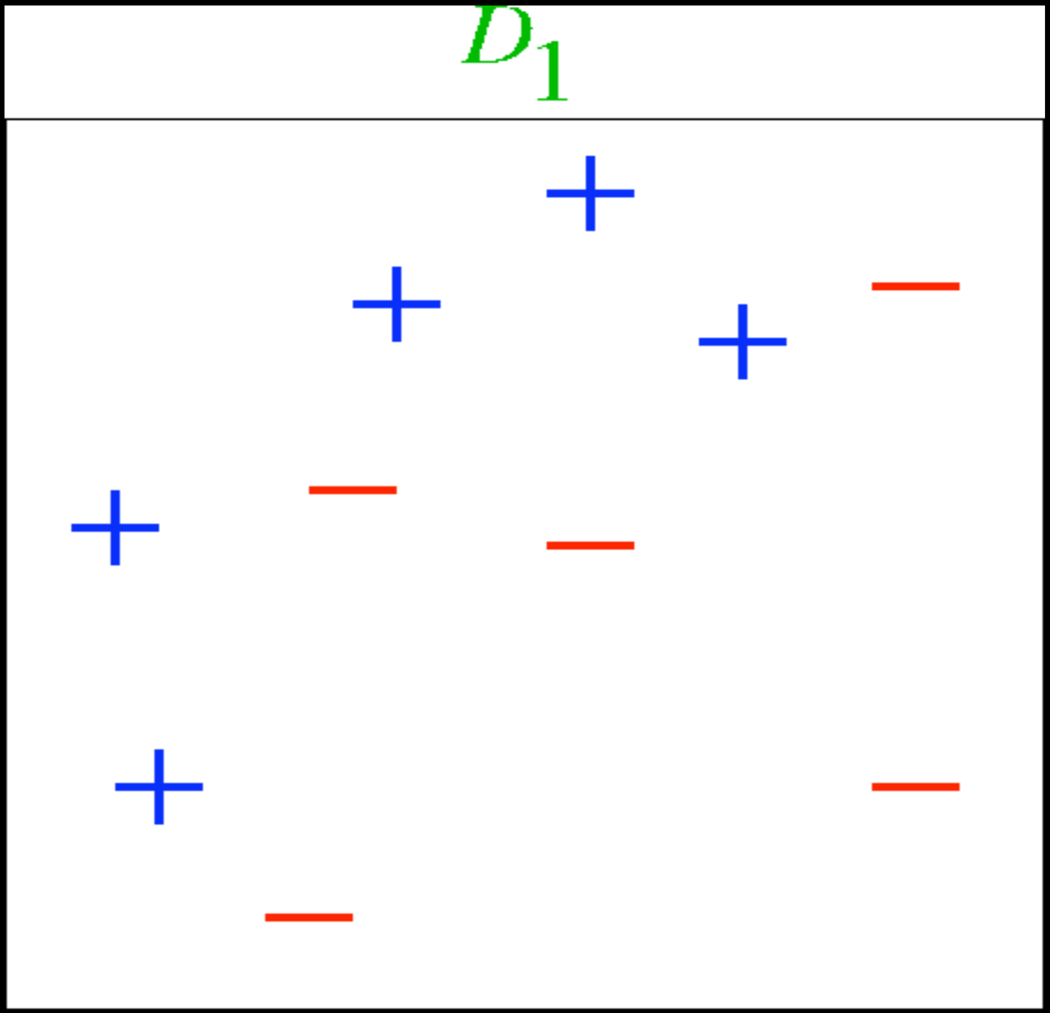
$$y_i = label(x_i); y_i \in \{-1, +1\}$$

boosting : introduction

- = Combine more classifiers in a master one
- Given
  - Labeled data set (training set)
  - Access to "weak learner" (error less than 50%)
- LOOP
  - Select weak learner
  - Concentrate on the hard (wrong classified) instances

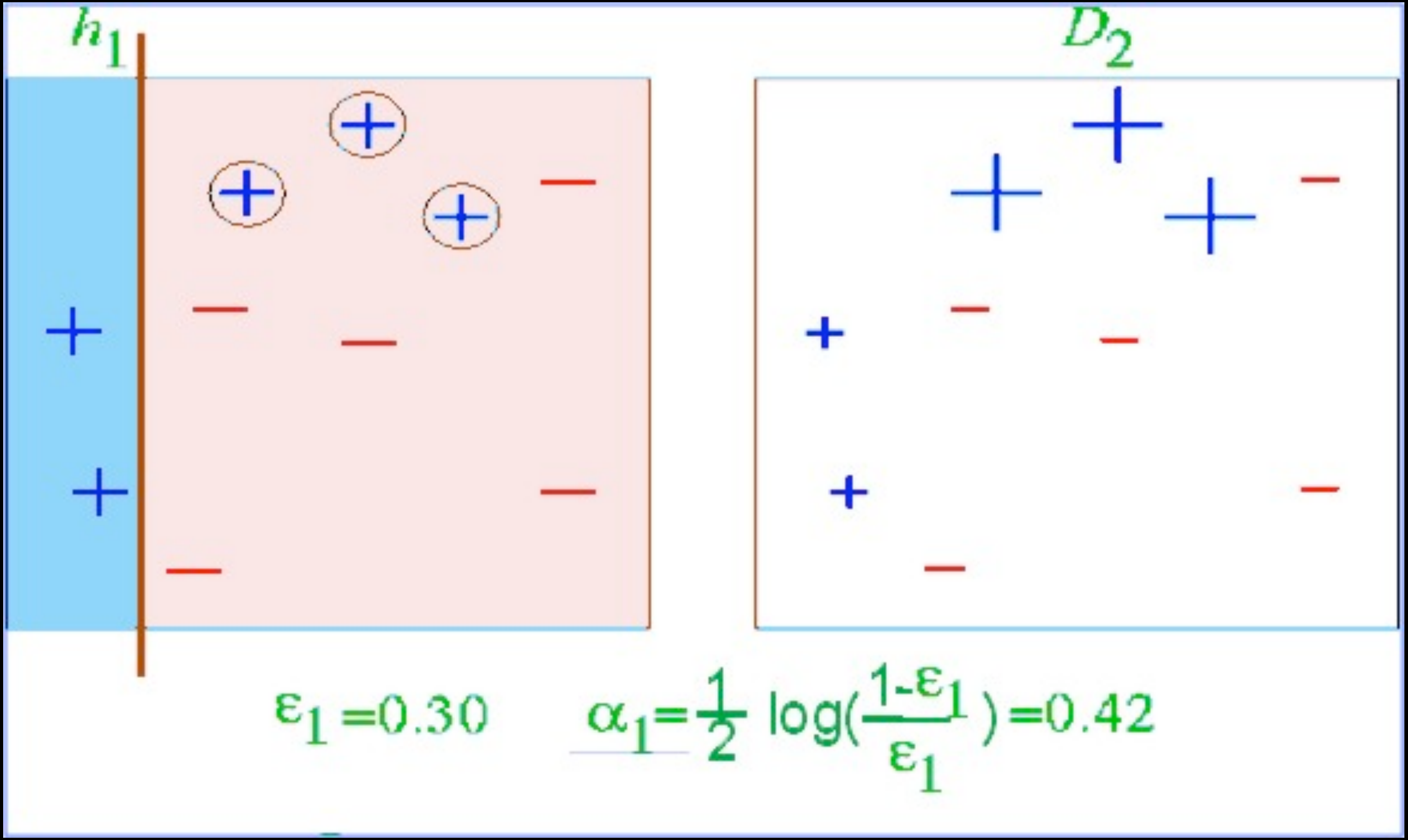


boosting example



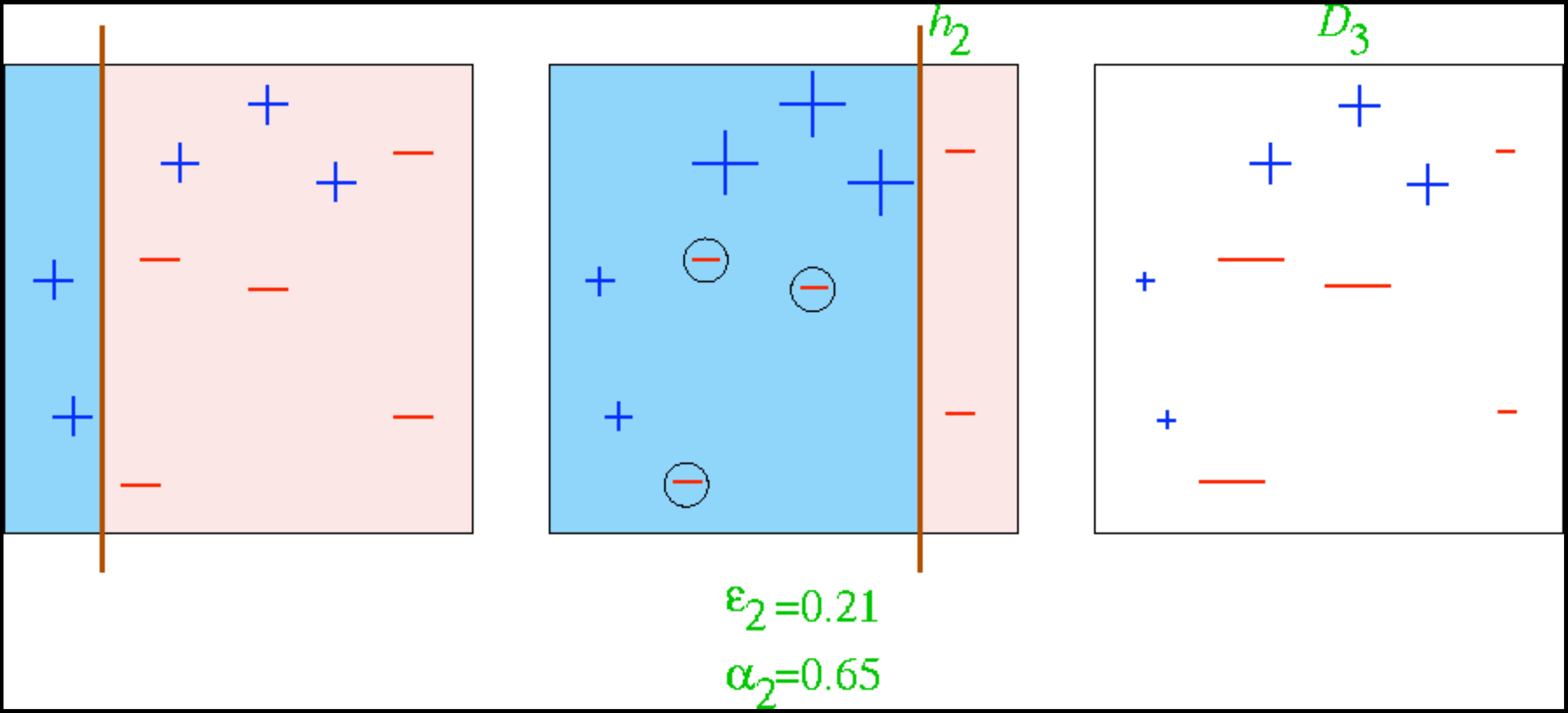
- Start with uniform distribution on data
- Weak learners = halfplanes

round 1

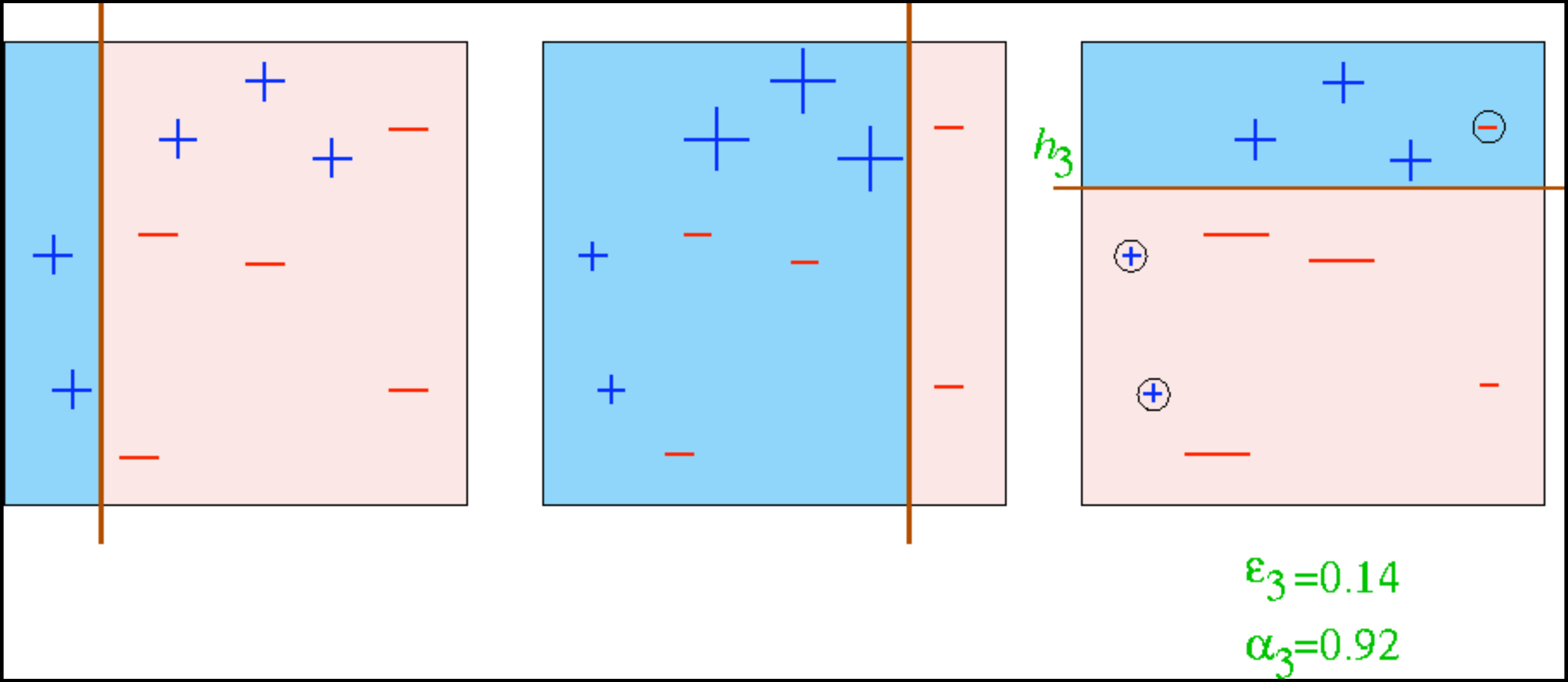


$\varepsilon_1 = 0.30$ $\alpha_1 = \frac{1}{2}\log\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right) = 0.42$

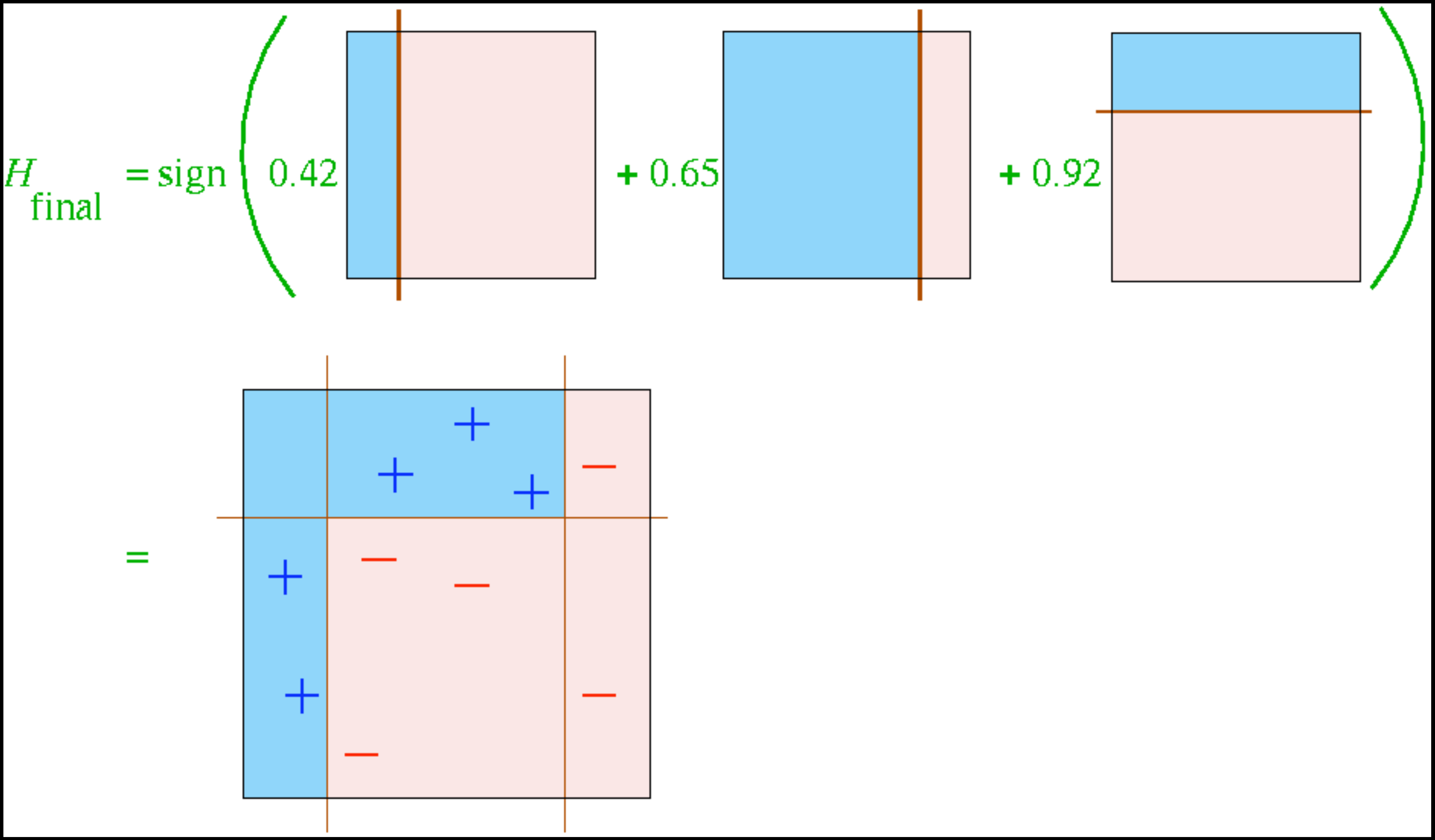
round 2



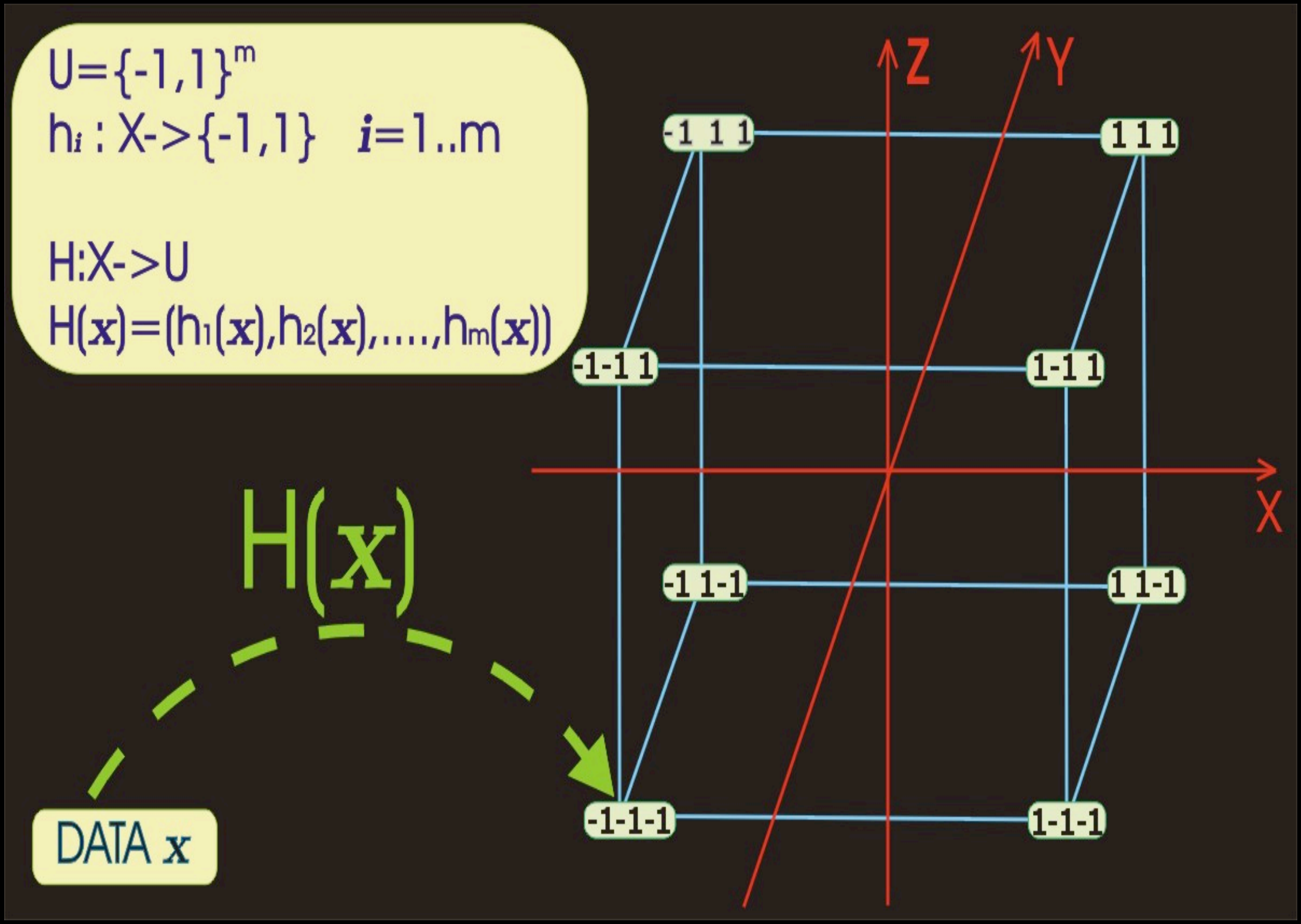
round 3



final hypothesis


$H_{\text{final}} = \text{sign}\left(0.42 \;\; + 0.65 \;\; + 0.92\right)$
=
+
−

the hypothesis points space



$U=\{-1,1\}^m$
$h_i : X \to \{-1,1\} \quad i=1..m$
$H: X \to U$
$H(\mathbf{x})=(h_1(\mathbf{x}),h_2(\mathbf{x}),\ldots,h_m(\mathbf{x}))$
$H(x)$
DATA x
Z
Y
X
-1 1 1
1 1 1
-1-1 1
1-1 1
-1 1-1
1 1-1
-1-1-1
1-1-1

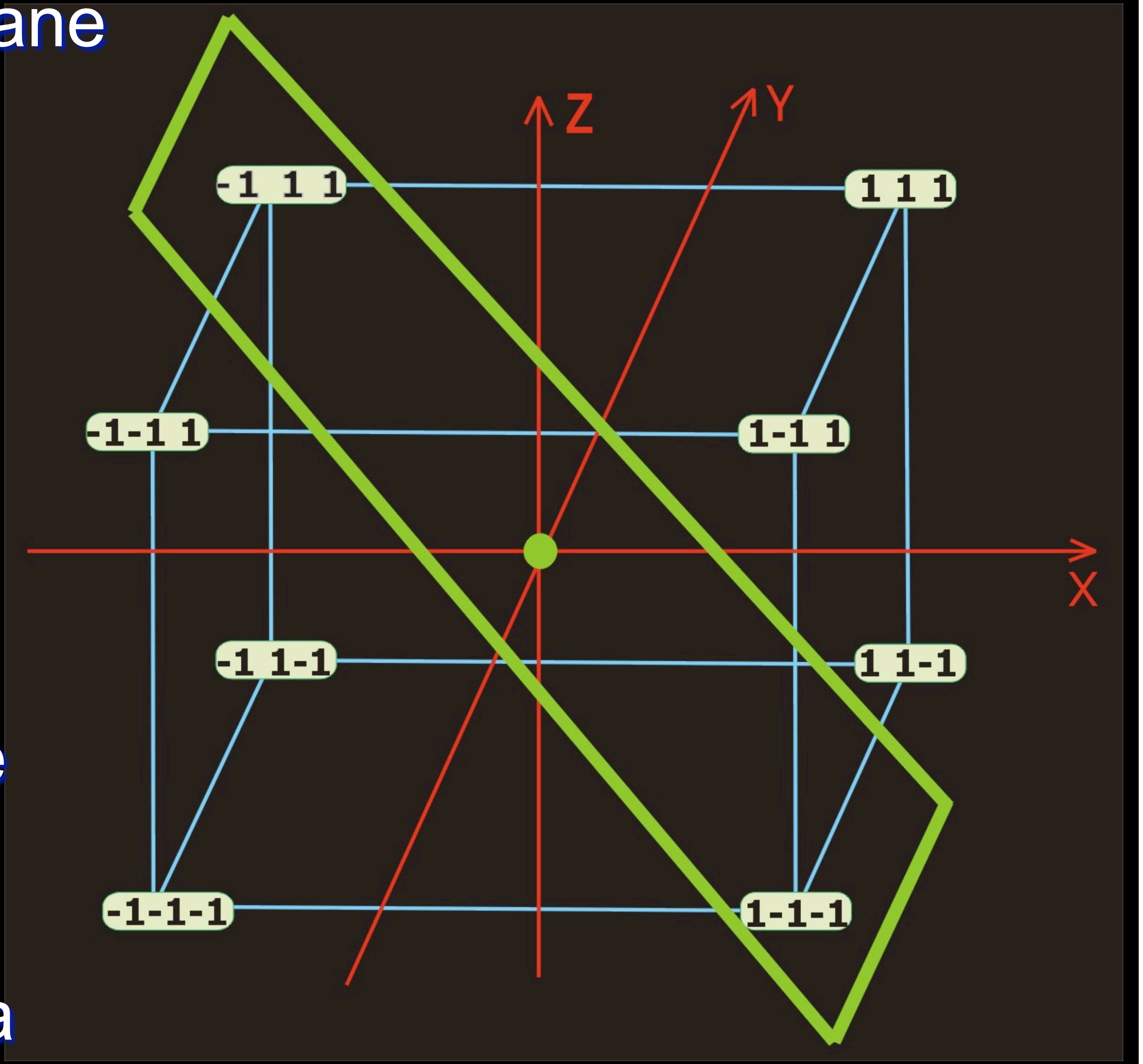
- Mapping from data space to hypothesis space
- The Hyper cube in hypothesis space
- Decision stumps

separation

- Separation hyperplane

$$F(x) = \sum_t \alpha_t h_t(x)$$

$$H_{\text{FIN}}(x) = \operatorname{sgn}\big(F(x)\big)$$



- Geometric interpretation
- Optimal hyperplane
- Non-separable data

AdaBoost - technical

- Start with uniform distrib D_1: $D_1(i) = \frac{1}{m}$
- At every round t=1 to T
  - given D_t
    - find weak hypothesis $h_t : X \longrightarrow \{-1,1\}$
    - with error $\varepsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i]$
    - compute "belief" in h_t $\boxed{\alpha_t = \frac{1}{2} \ln \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right) > 0}$
    - update distribution $D_{t+1} = \frac{D_t}{Z_t} \cdot \begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \end{cases}$
- final hypothesis:

$$H_{\text{final}}(x) = \text{sgn}\left(\sum_t \alpha_t h_t(x) \right)$$

a bayesian interpretation

- Bayes decision rule : predict 1 if

$$\Pr\left[y=1 \mid h_1(x),\ldots,h_T(x)\right] > \Pr\left[y=0 \mid h_1(x),\ldots,h_T(x)\right],$$

- For Adaboost : predict 1 if

$$\Pr\left[y=1\right] \prod_{t:h_t(x)=0} \epsilon_t \prod_{t:h_t(x)=1} (1-\epsilon_t) > \Pr\left[y=0\right] \prod_{t:h_t(x)=0} (1-\epsilon_t) \prod_{t:h_t(x)=1} \epsilon_t,$$

$$H_{\text{final}}(x) = \operatorname{sgn}\left(\sum_t \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) * h_t(x)\right)$$

AdaBoost – update rule

- Start with uniform distrib D_1 : $D_1(i) = \frac{1}{m}$
- At every round t=1 to T
  - given D_t
    - find weak hypothesis $h_t : X \rightarrow \{-1, 1\}$
    - with error $\varepsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i]$
    - compute "belief" in h_t $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) > 0$
    - update distribution

$$D_{t+1} = \frac{D_t}{Z_t} \cdot \begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \end{cases}$$

- final hypothesis:

$$H_{\text{final}}(x) = \text{sgn}\left(\sum_t \alpha_t h_t(x)\right)$$

online allocation - hedge algorithm

- Parameters

  $\beta \in [0,1]$; N strategies (systems)

  initial weights $\mathrm{w}^1 \in [0,1]^N ; \sum_{i=1}^{N} w_i^1 = 1$

- LOOP for episode t=1,2,…,T
  - Choose allocation $p_i^t = \frac{w_i^t}{\sum_{i=1}^{N} w_i^t}$
  - Receive loss vector $l^t \in [0,1]^N$
  - Suffer loss $p^t * l^t$
  - Update weights $w_i^{t+1} = w_i^t \bullet \beta^{l_i^t}$

- Hedge loss $L_{HEDGE} = \sum_{t=1}^{T} p^t * l^t$

- …not “too much worse” than the best system

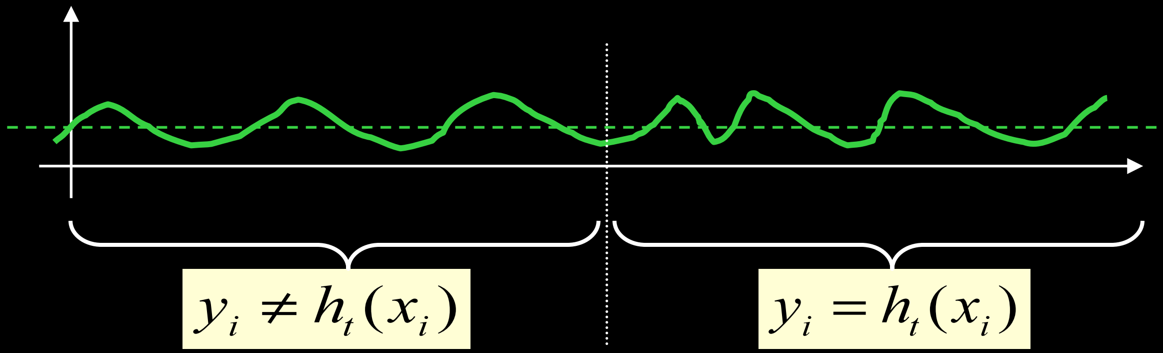
$$L_{HEDGE} \le \frac{\ln\left(\frac{1}{\beta}\right) L_{SYSTEM} + \ln N}{1-\beta}$$

AdaBoost – distribution update

$$h_t : X \rightarrow \{-1, 1\}$$

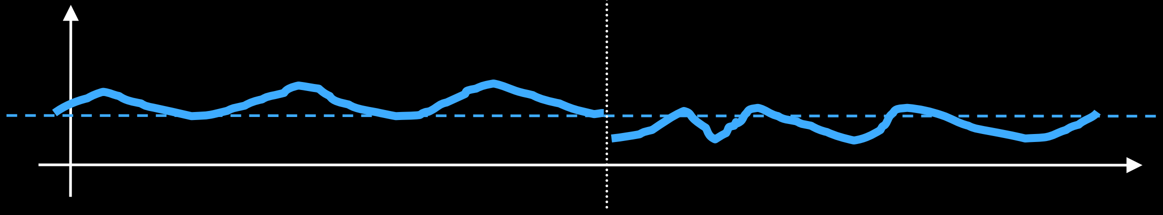
$$\varepsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i]$$

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right) > 0$$



$$D_{t+1} = \frac{D_t}{Z_t} \cdot e^{\alpha_t}$$

$$D_{t+1} = \frac{D_t}{Z_t} \cdot e^{-\alpha_t}$$



- “neutralize” the last weak hypothesis

training error

- Theorem

[Freund&Schapire '97]

$$\text{training error}(H_{\text{final}}) \le 2^T \prod_t \sqrt{\varepsilon_t(1-\varepsilon_t)}$$

proof

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

$$D_{t+1} = \frac{D_t}{Z_t}\cdot\begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \ne h_t(x_i)\end{cases} = \frac{D_t}{Z_t} e^{-\alpha_t y h_t(x)}$$

$$F(x) = \sum_t \alpha_t h_t(x)$$

$$H_{\text{FIN}}(x) = \operatorname{sgn}(F(x))$$

$$Z_t = \sum_x D_t(x)\exp(-y_x\alpha_t h_t(x))$$

$$= \sum_{y_x h_t(x)=1} D_t(x)\exp(-\alpha_t) + \sum_{y_x h_t(x)=-1} D_t(x)\exp(+\alpha_t)$$

$$= \exp(-\alpha_t)(1-\varepsilon_t) + \exp(\alpha_t)\varepsilon_t$$

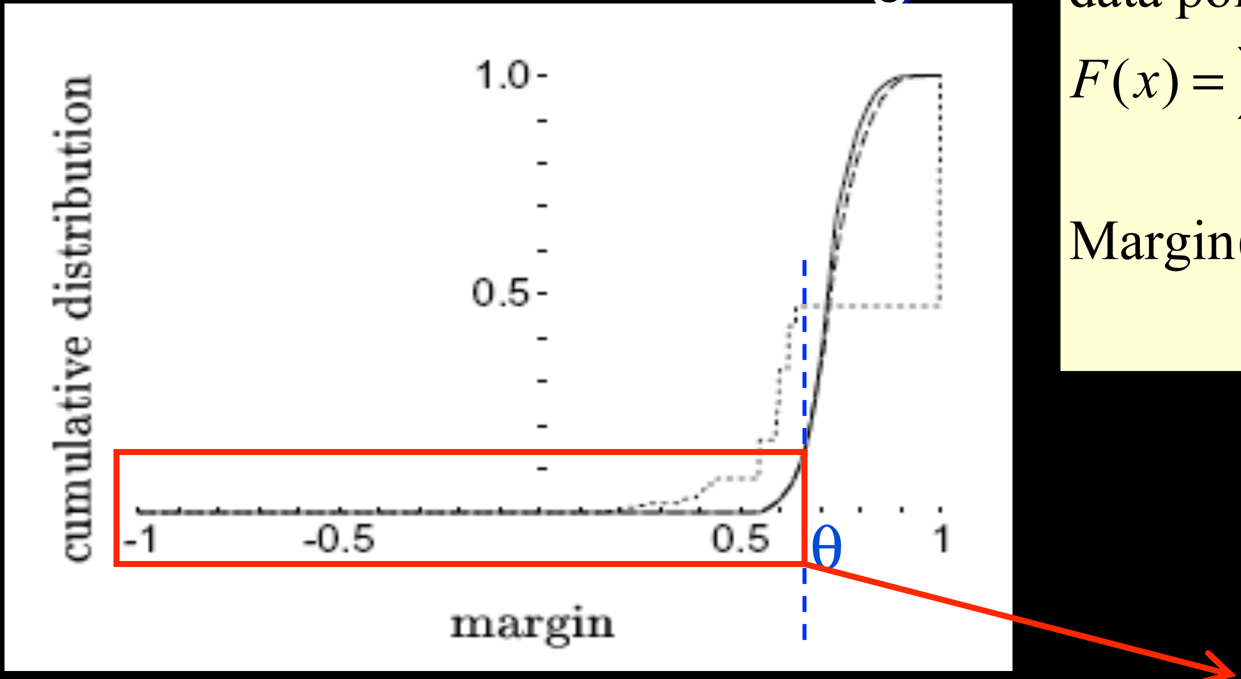
$$= (1-\varepsilon_t)\sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}} + \varepsilon_t\sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}} = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$$

$$D_{T+1}(x) = \frac{1}{m}\prod_t \frac{\exp(-y\alpha_t h_t(x))}{Z_t} = \left(\frac{1}{\prod_t Z_t}\right)\frac{1}{m}\exp(-yF(x))$$

$$\frac{1}{m}\sum_x \exp(-yF(x)) = \prod_t Z_t$$

Boosting and margin distribution

- Adaboost maximizes margins



data point $x \in X$;

$$F(x) = \sum_t \alpha_t * h_t(x)$$

$$\text{Margin}(x) = y(x) * \frac{F(x)}{\sum_t \alpha_t} \in [-1,1]$$

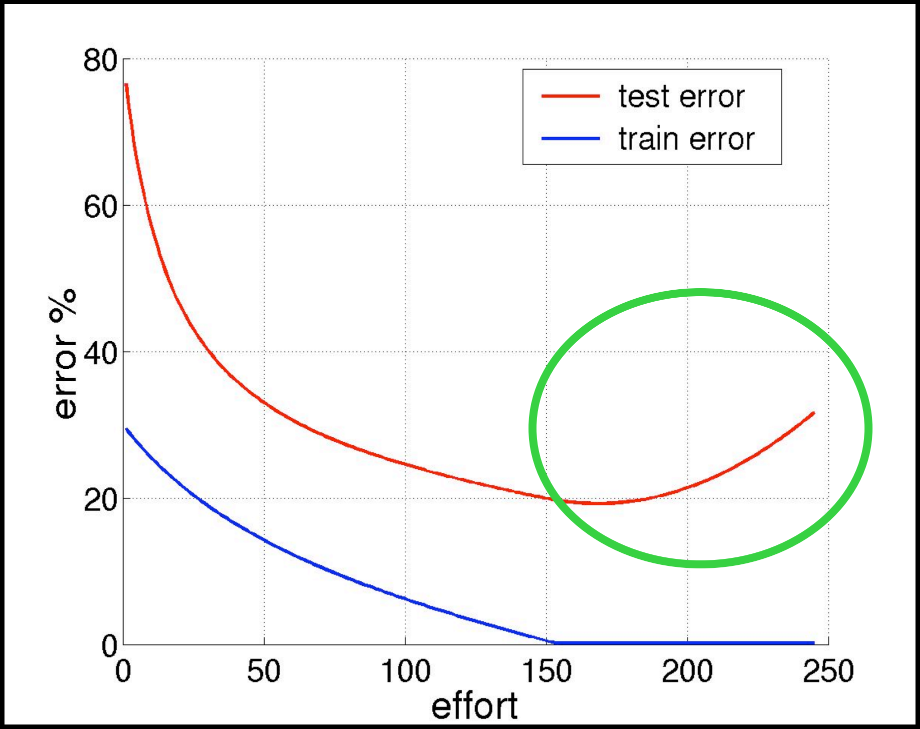
- Schapire,Freund,Barlett,Lee '98
  - For any 0<θ<1

training miss-confidence

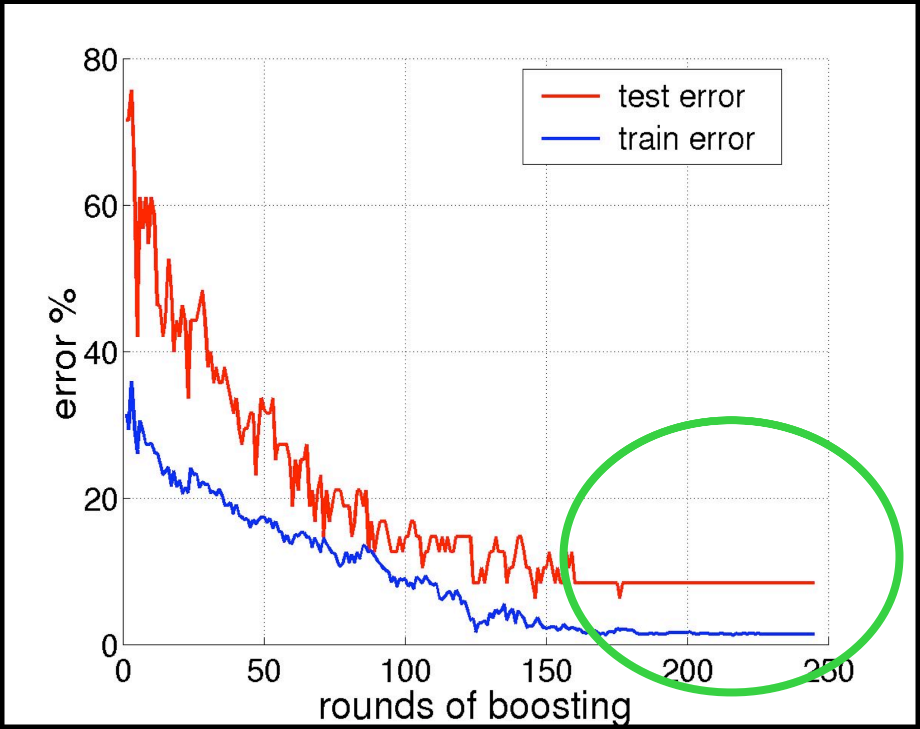
$$\boxed{\mathbf{P}_{(x,y)\sim S}\left[yf(x) \le \theta\right]} \le 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}.$$

Why margins are important

typical curve



AdaBoost



- Generalization error

generalization error- based on margins

- Schapire,Freund,Barlett,Lee 1998

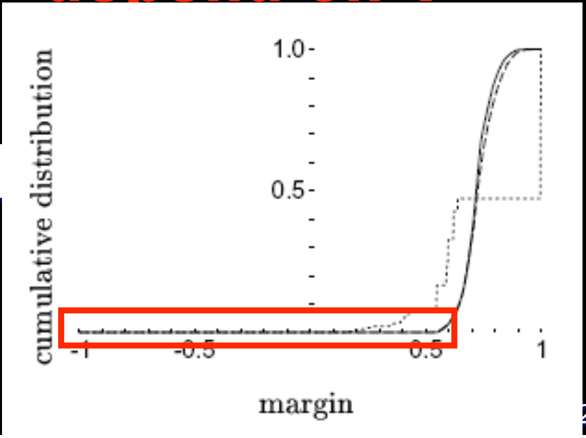
Theorem 1 *Let $\mathcal{D}$ be a distribution over $X \times \{-1, 1\}$, and let S be a sample of m examples chosen independently at random according to $\mathcal{D}$. Assume that the base-classifier space $\mathcal{H}$ is finite, and let $\delta > 0$. Then with probability at least $1 - \delta$ over the random choice of the training set S, every weighted average function $f \in \mathcal{C}$ satisfies the following bound for all $\theta > 0$:*

$$\boxed{\mathbf{P}_{\mathcal{D}}\left[yf(x) \le 0\right]} \le \boxed{\mathbf{P}_S\left[yf(x) \le \theta\right]} + O\boxed{\left(\frac{1}{\sqrt{m}}\left(\frac{\log m \log|\mathcal{H}|}{\theta^2} + \log(1/\delta)\right)^{1/2}\right)}.$$

testing error **training miss-confidence** **does not depend on T**

- Original generalization bound Schap

$$\text{testing\_error} \le \text{training\_error} + \mathrm{O}\left(\sqrt{\frac{Td}{m}}\right)$$



AdaBoost - remarks

- AdaBoost is adaptive

  $$\gamma = \min\left(\frac{1}{2} - \varepsilon_t\right)$$

  - does not need to know γ or T a priori
  - can exploit $\varepsilon_t \ll \frac{1}{2} - \gamma$

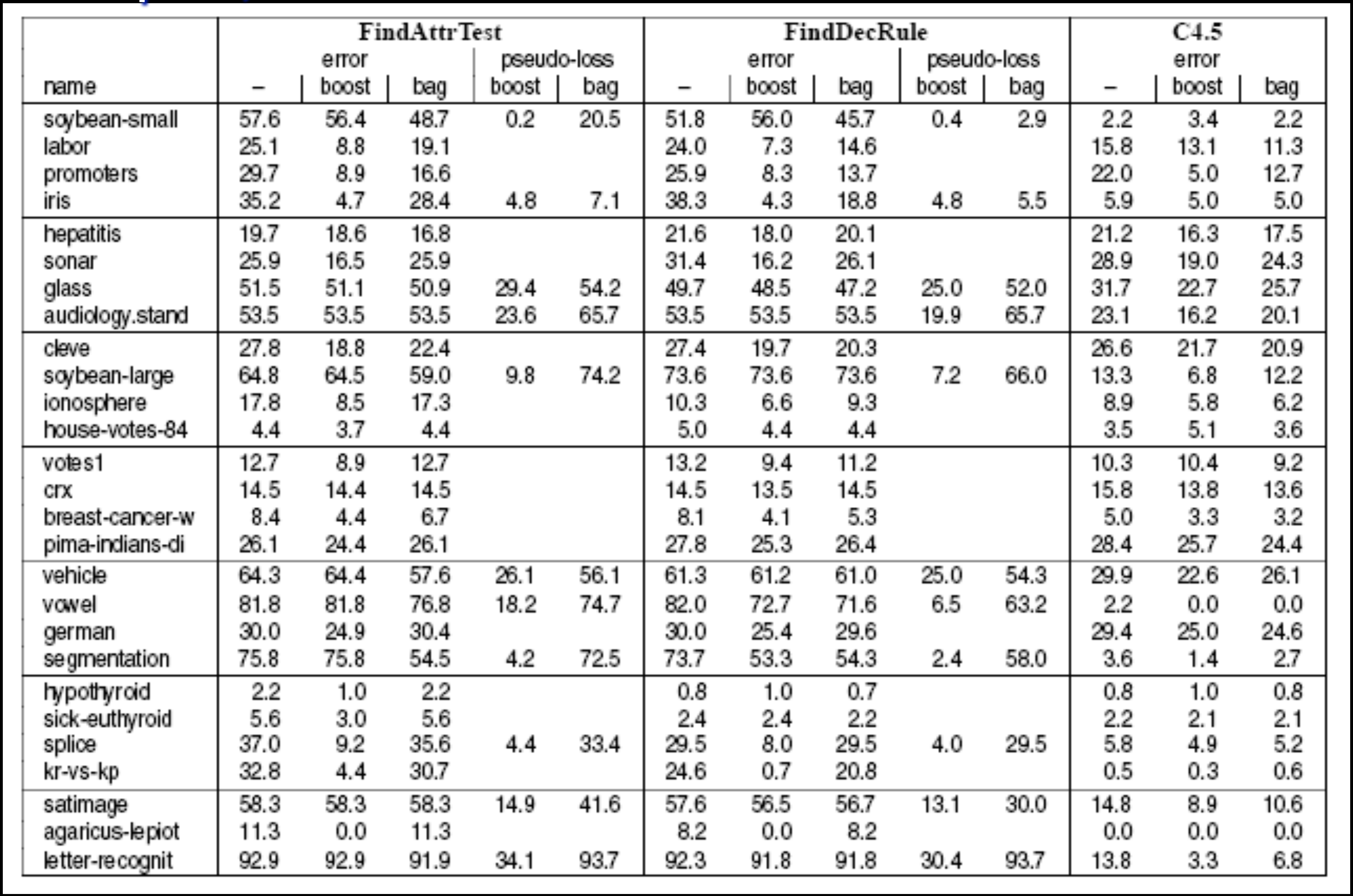
- GOOD : does not overfit
- BAD : Susceptible to noise
  - but a nice property : identify outliers

Recent advances

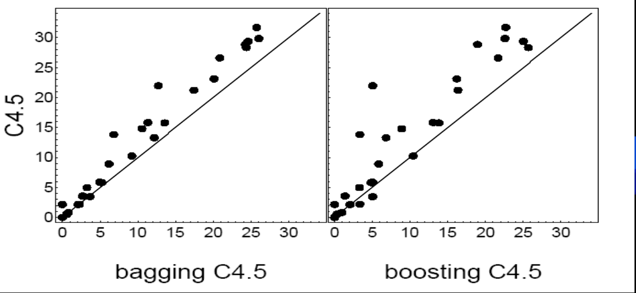
- Normalized weights - Hyperplane is always convergent
- If data is nonseparable weight vector is convergent
- If data is separable the weights are cyclic

Experiments with a New Boosting Algorithm

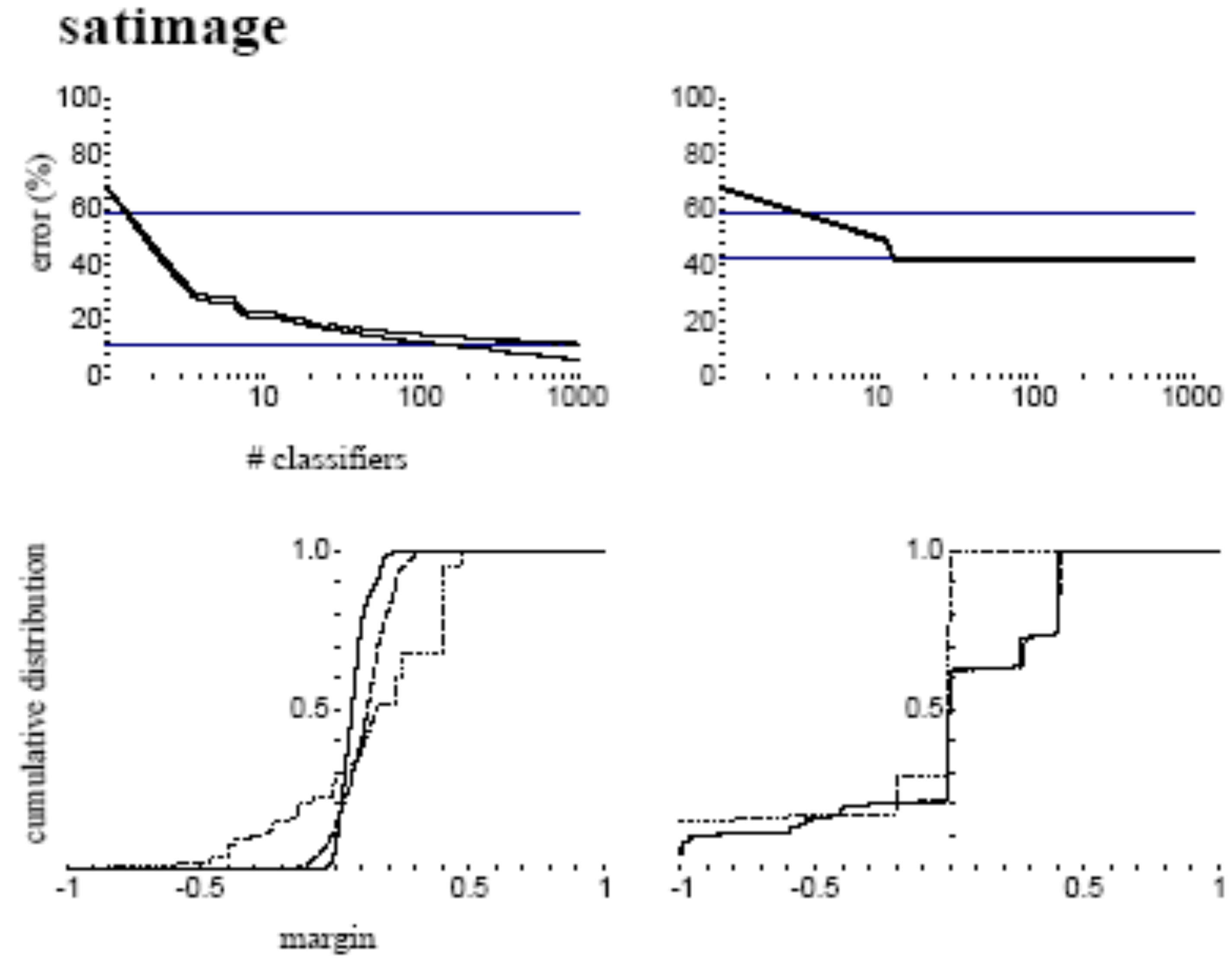
- Schapire,Freund 1996



	FindAttrTest					FindDecRule					C4.5		
	error			pseudo-loss		error			pseudo-loss		error		
name	–	boost	bag	boost	bag	–	boost	bag	boost	bag	–	boost	bag
soybean-small	57.6	56.4	48.7	0.2	20.5	51.8	56.0	45.7	0.4	2.9	2.2	3.4	2.2
labor	25.1	8.8	19.1			24.0	7.3	14.6			15.8	13.1	11.3
promoters	29.7	8.9	16.6			25.9	8.3	13.7			22.0	5.0	12.7
iris	35.2	4.7	28.4	4.8	7.1	38.3	4.3	18.8	4.8	5.5	5.9	5.0	5.0
hepatitis	19.7	18.6	16.8			21.6	18.0	20.1			21.2	16.3	17.5
sonar	25.9	16.5	25.9			31.4	16.2	26.1			28.9	19.0	24.3
glass	51.5	51.1	50.9	29.4	54.2	49.7	48.5	47.2	25.0	52.0	31.7	22.7	25.7
audiology.stand	53.5	53.5	53.5	23.6	65.7	53.5	53.5	53.5	19.9	65.7	23.1	16.2	20.1
cleve	27.8	18.8	22.4			27.4	19.7	20.3			26.6	21.7	20.9
soybean-large	64.8	64.5	59.0	9.8	74.2	73.6	73.6	73.6	7.2	66.0	13.3	6.8	12.2
ionosphere	17.8	8.5	17.3			10.3	6.6	9.3			8.9	5.8	6.2
house-votes-84	4.4	3.7	4.4			5.0	4.4	4.4			3.5	5.1	3.6
votes1	12.7	8.9	12.7			13.2	9.4	11.2			10.3	10.4	9.2
crx	14.5	14.4	14.5			14.5	13.5	14.5			15.8	13.8	13.6
breast-cancer-w	8.4	4.4	6.7			8.1	4.1	5.3			5.0	3.3	3.2
pima-indians-di	26.1	24.4	26.1			27.8	25.3	26.4			28.4	25.7	24.4
vehicle	64.3	64.4	57.6	26.1	56.1	61.3	61.2	61.0	25.0	54.3	29.9	22.6	26.1
vowel	81.8	81.8	76.8	18.2	74.7	82.0	72.7	71.6	6.5	63.2	2.2	0.0	0.0
german	30.0	24.9	30.4			30.0	25.4	29.6			29.4	25.0	24.6
segmentation	75.8	75.8	54.5	4.2	72.5	73.7	53.3	54.3	2.4	58.0	3.6	1.4	2.7
hypothyroid	2.2	1.0	2.2			0.8	1.0	0.7			0.8	1.0	0.8
sick-euthyroid	5.6	3.0	5.6			2.4	2.4	2.2			2.2	2.1	2.1
splice	37.0	9.2	35.6	4.4	33.4	29.5	8.0	29.5	4.0	29.5	5.8	4.9	5.2
kr-vs-kp	32.8	4.4	30.7			24.6	0.7	20.8			0.5	0.3	0.6
satimage	58.3	58.3	58.3	14.9	41.6	57.6	56.5	56.7	13.1	30.0	14.8	8.9	10.6
agaricus-lepiot	11.3	0.0	11.3			8.2	0.0	8.2			0.0	0.0	0.0
letter-recognit	92.9	92.9	91.9	34.1	93.7	92.3	91.8	91.8	30.4	93.7	13.8	3.3	6.8



boosting vs bagging



Boosting vs SVMs

- Map data in Feature space
  - Adaboost – weak hypotesis
  - SVM – kernel function
- Separate with a hyperplane
- Maximize margins
  - AdaBoost – iterative convex optimization
  - SVM – global optimization via Lagrange multipliers

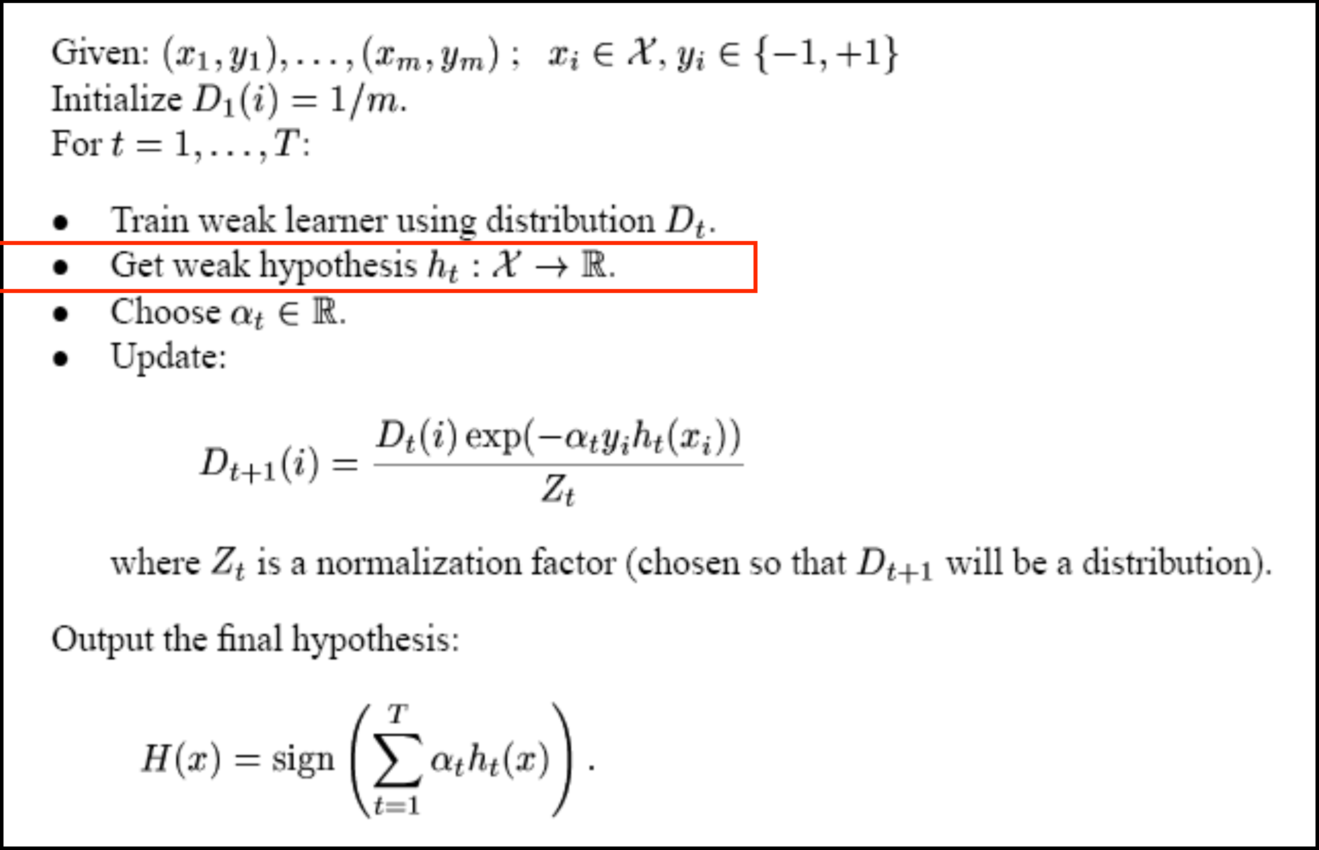
- Different norms can result in very different margins
  - AdaBoost – l1 norm
  - SVM – l2 norm
- The computation requirements are different
  - AdaBoost – linear programming
  - SVM – quadratic programming
- A different approach for searching efficiently in high dimensional space
  - AdaBoost – greedy
  - SVM – Kernel method

☺...hope you are still with me

- boosting and on-line learning
- AdaBoost algorithm

- extensions
- applications

boosting using confidence-rated predictions



Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : \mathcal{X} \to \mathbb{R}$.
- Choose $\alpha_t \in \mathbb{R}$.
- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

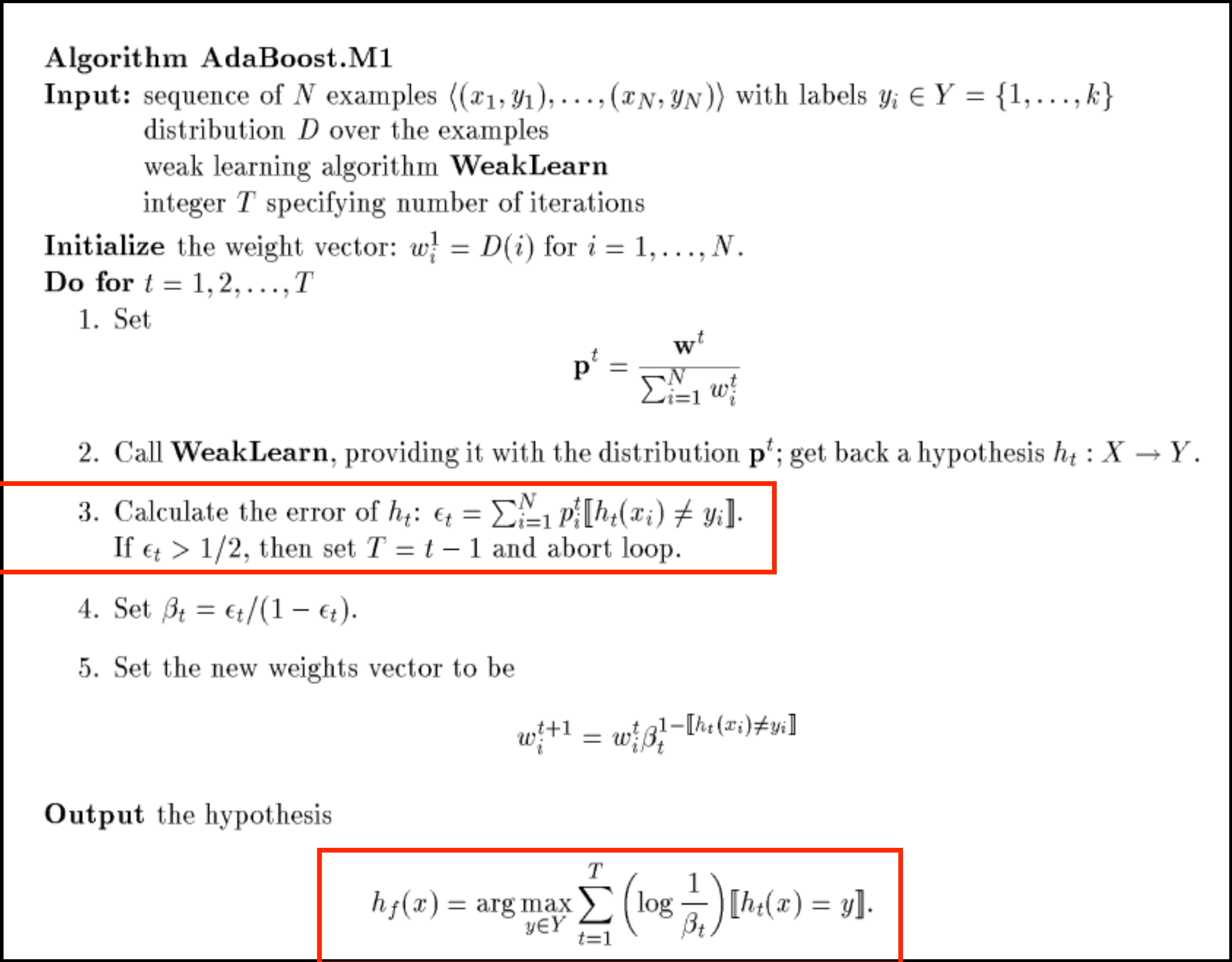
$$u_i = y_i h_t(x_i)$$

$$r = \sum_i D(i) u_i.$$

$$\begin{aligned} Z &= \sum_i D(i) e^{-\alpha u_i} \\ &\leq \sum_i D(i) \left(\frac{1 + u_i}{2} e^{-\alpha} + \frac{1 - u_i}{2} e^{\alpha} \right). \end{aligned}$$

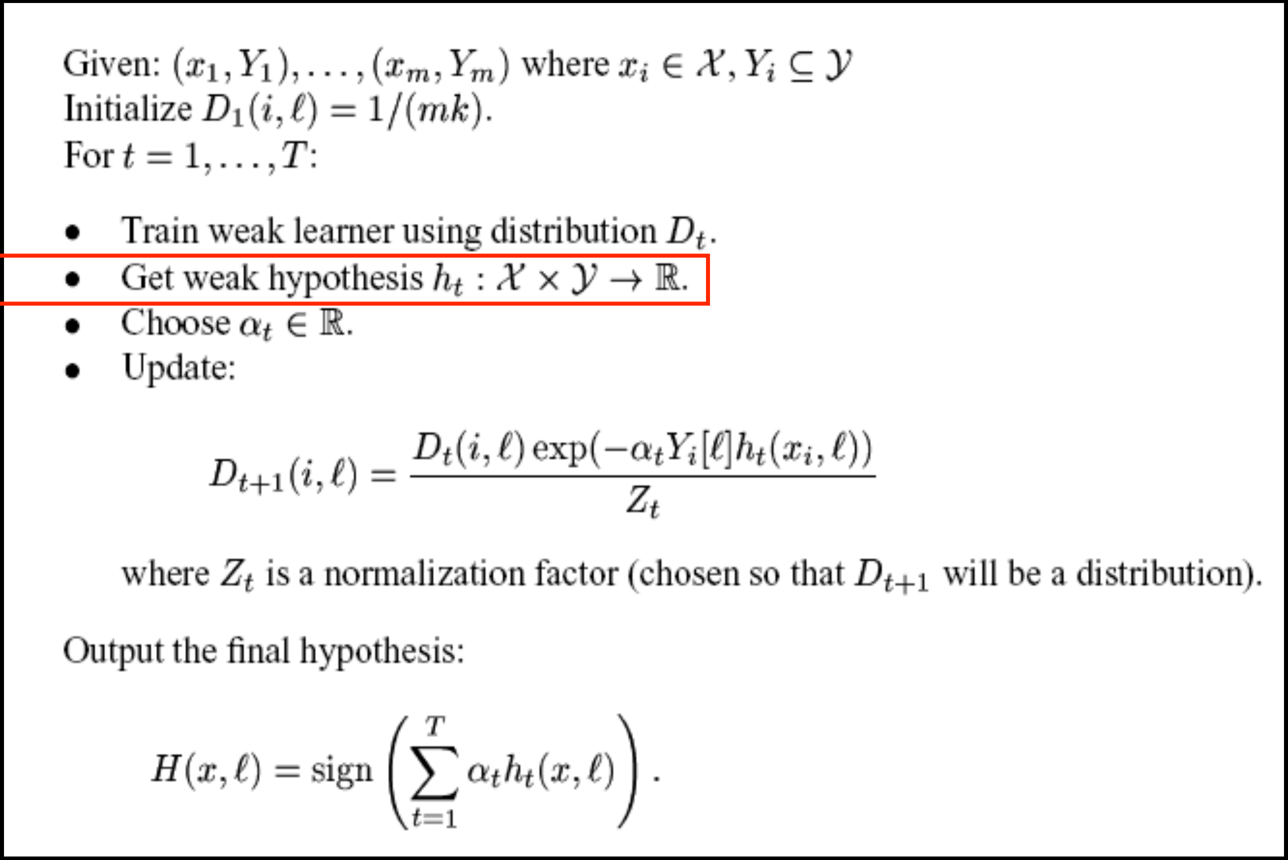
$$\alpha = \tfrac{1}{2} \ln \left(\frac{1 + r}{1 - r} \right)$$

multiclass : AdaBoost.M1



```
Algorithm AdaBoost.M1
Input: sequence of $N$ examples $\langle(x_1, y_1), \ldots, (x_N, y_N)\rangle$ with labels $y_i \in Y = \{1, \ldots, k\}$
       distribution $D$ over the examples
       weak learning algorithm WeakLearn
       integer $T$ specifying number of iterations
Initialize the weight vector: $w_i^1 = D(i)$ for $i = 1, \ldots, N$.
Do for $t = 1, 2, \ldots, T$
  1. Set
        $$\mathbf{p}^t = \frac{\mathbf{w}^t}{\sum_{i=1}^N w_i^t}$$
  2. Call WeakLearn, providing it with the distribution $\mathbf{p}^t$; get back a hypothesis $h_t : X \to Y$.
  3. Calculate the error of $h_t$: $\epsilon_t = \sum_{i=1}^N p_i^t [\![h_t(x_i) \neq y_i]\!]$.
     If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
  4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
  5. Set the new weights vector to be
        $$w_i^{t+1} = w_i^t \beta_t^{1 - [\![h_t(x_i) \neq y_i]\!]}$$
Output the hypothesis
        $$h_f(x) = \arg\max_{y \in Y} \sum_{t=1}^T \left(\log \frac{1}{\beta_t}\right) [\![h_t(x) = y]\!].$$
```

multiclass,multilabel : AdaBoost.MH



```
Given: $(x_1, Y_1), \ldots, (x_m, Y_m)$ where $x_i \in \mathcal{X}, Y_i \subseteq \mathcal{Y}$
Initialize $D_1(i, \ell) = 1/(mk)$.
For $t = 1, \ldots, T$:

  • Train weak learner using distribution $D_t$.
  • Get weak hypothesis $h_t : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.
  • Choose $\alpha_t \in \mathbb{R}$.
  • Update:

        $D_{t+1}(i, \ell) = \frac{D_t(i, \ell) \exp(-\alpha_t Y_i[\ell] h_t(x_i, \ell))}{Z_t}$

    where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).

Output the final hypothesis:

    $H(x, \ell) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x, \ell)\right)$.
```

- Hamming Loss
- Decompose in k binary problems

$$\frac{1}{k} \mathrm{E}_{(x,Y)\sim D} [\, |h(x) \,\Delta\, Y| \,]$$

$$H : \mathcal{X} \to 2^{\mathcal{Y}}$$

output coding for multiclass problems

	L1	L2	L3	L4		P1	P2
a	0	0	0			1	0
b	0	0	1			0	1
c	0	1	0			0	0
		1					
y	1	0	0				1
Z	1	0	1				0
z	1	1	0				1

- Every learner trained on a subset of labels
- Label is identified by all predictions, as numbers in bynary
- Error-check redundancy learners

InfoBoost – [Aslam '2000]

Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

1. Train weak learner using distribution D_t.
2. Get weak hypothesis $h_t : \mathcal{X} \to \mathbb{R}$.
3. Choose $\alpha_t[-1] \in \mathbb{R}$ and $\alpha_t[+1] \in \mathbb{R}$;

$$\boxed{\text{let } \alpha_t(z) = \begin{cases} \alpha_t[-1] & \text{if } z < 0, \\ \alpha_t[+1] & \text{if } z \geq 0. \end{cases}}$$

4. Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t(h_t(x_i)) y_i h_t(x_i))}{Z_t}$$

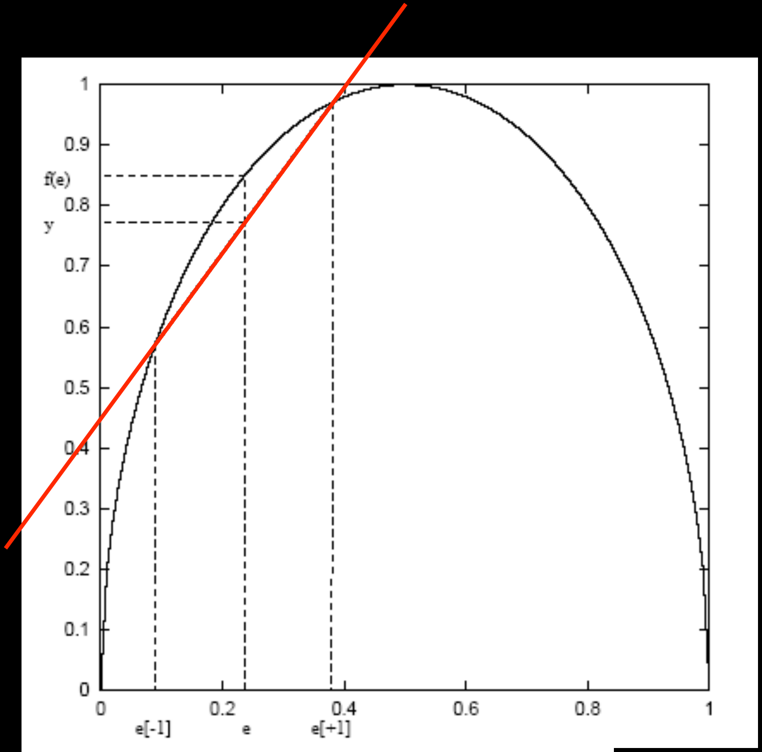
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t(h_t(x)) h_t(x) \right).$$

h_1	-1	$+1$
-1	$5/16$	$3/16$
$+1$	$1/16$	$7/16$

h_2	-1	$+1$
-1	$6/16$	$2/16$
$+1$	$2/16$	$6/16$



function $2\sqrt{x(1-x)}$.

applications

- Boostexter [Schapire,Singer '2000]
  - Multilabel,multiclass text categorization
  - Based on AdaBoost.MH
- auction price uncertainty ATTac-2001
  - [Schapire,Stone,McAllester,Littman,Csirik 02]
  - Reduce the problem to multiclass, multilabel setup
  - AdaBooost.MH
- RankBoost[Iyer,Lewis,Schapire,Singer,Singhal 2000]
  - Ranking instead of classification
  - IR routing

SO

- AdaBoost has many advantages. At least in theory…
  - fast, simple, easy to program
  - no parameters to tune
  - no prior knowledge about the weak learner
  - theoretical guarantees
  - weak learners only need to be better than random
- In practice
  - Weak hypothesis are closer and closer to “random”
  - It <u>may</u> overfit
  - Theoretical guarantees are loose
  - Real data is not fully separable
  - Performance depends both on data and weak learner